# Thomas Jones – CS5567-0002 – Project 3

Quick Note: All these were run on a Mac Studio M1 using MPS support.

## Object Detection with YOLO8

A graph of speed and speed

Description automatically generated

A graph of a speed line

Description automatically generated with medium confidence

A graph of different colors and numbers

Description automatically generated

A graph of different colors and sizes

Description automatically generated

A graph of a graph

Description automatically generated with medium confidenceA group of graphs showing loss

Description automatically generated with medium confidence

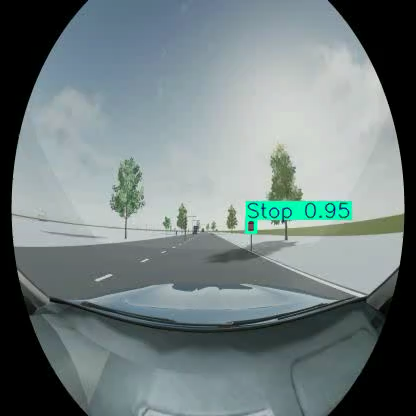
A graph of a graph of a training

Description automatically generated with medium confidence

A screenshot of a video camera

Description automatically generated

Double click to play MP4



## Segmentation Results

### UUNet

Due to upgrades for Tensorflow the error function needed to be slightly rewritten to no longer use deprecated methods. In all 5 different configurations were tested (see spreadsheet) with most results meeting expected results. It was found that batch size adjustments had the most impact with the best DICE score being 0.902 or a 90.2% coverage match vs the test masks. This was achieved with a batch size of 15 over a total of 100 epochs though the best epoch against validation loss was at 93.

A graph of training and validation

Description automatically generated

This produced the following test images (only images with highlighted areas shown for brevity)

A white mask on a black background

Description automatically generated

A white mask on a black background

Description automatically generated

A white mask on a black background

Description automatically generated

A white mask on a black background

Description automatically generated

A white mask on a black background

Description automatically generated

A white outline of a mask

Description automatically generated

### SegFormer

The SegFormer image segmentation ended up being challenging for some deep technical issues with Pytorch, specifically M1 does not support float64 so results were inconsistent with CPU or CUDA results. See <https://github.com/Lightning-AI/pytorch-lightning/discussions/15407> for a quick reference. As a result these were run on the CPU which then produced the correct results. While a this was a difficulty for the experimenter (who could have used Colab) it does speak to some of the broader consumer challenges that might be faced in the future as well as for future experimenters.

Three random images were selected, shown with corresponding masks.

A bathroom with a sink and a window

Description automatically generated

A bathroom with a tub sink and a window

Description automatically generated

Comparing the data against the mask:

A graph of a fish and a hook

Description automatically generated with medium confidence

We can see that the results reasonably match the masks provided. The fine tuning produced a mean IOU of 0.530, a mean accuracy of 0.818, and an overall accuracy of 0.982.

### Segmentation Discussion

While similar in appearance the two networks performed different tasks in different manners. A big constraint on direct comparison was the limited size of the SegFormer dataset. Without this effective parameter changes were not feasible so the observation that UUNet, for the image set given, performed better with smaller batches, cannot be compared. In terms of geometric structures, SegFormer appears to be a better choice against non-organic or more regular shapes. UUNet operated well (higher metrics) against a task that was more structure filling in nature, i.e. filling out the area of a possible tumor, hence a more organic shape. Neither system seemed to have any issues with overlapping structures. Both networks were trained for 100 epochs using the Adam optimizer with default parameters (lr=0.002, etc).

## Transfer Learning

The same additional architecture was used for both networks, specifically a resizing layer at the head and the addition of a batch normalization and dense layer at the tail. The activation function in both cases was switched to softmax from relu. Additionally, initial experimentation found that tanh worked better as an activation for the dense layer, it is possible this helped to exclude labels where two labels were close though further experimentation would be necessary.

### ResNet50

The training performance showed an expected initial sharp uptick in accuracy with a tailing plateau. The validation accuracy followed a similar path eventually converging.

A graph of a graph

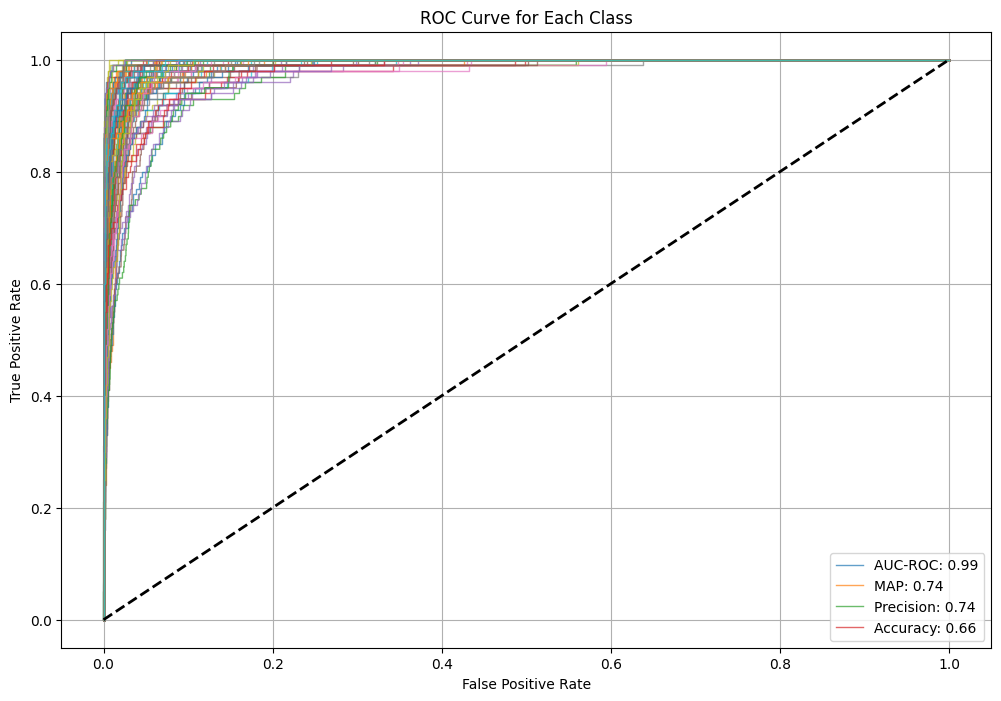
Description automatically generated

The loss curve shows similar results though as expected, training loss remained well above validation loss which remained relatively stable. While this might be an indicator of overfitting the convergence of the validation accuracy and lack of subsequent drop indicates a local minima was likely reached and little further training helped.

A graph of training and validation

Description automatically generated

The ROC curves for each of the classes show that the classification overall performed well given that the original images are only 32x32.



Looking at a few actual results,

A collage of different animals

Description automatically generated

It is at least visually clear that, while there were misses, the misses are actually reasonable mistakes. It is possible, for example, that the first image *could*be a road and the cloud does look like a sea. The butterfly, at this resolution, does appear skunk-like as well.

### VGG19

The accuracy characteristics for this network show what we would normally expect from training a network with a clear gap between training and validation accuracy.

A graph of a graph

Description automatically generated

While the loss curves to indicate an increase in validation error (though not necessarily shown in the accuracy) as training continued. While not completely indicative of overtraining it is clear that fewer epochs might have been required to achieve the results.

A graph of training and validation

Description automatically generated

The ROC curves for the various classes indicate accurate classification considering the resolution of the source images.

A graph of a curve

Description automatically generated

The predicted test labels are reasonable considering the resolution of the images. Apple was missed though it’s possible that was overtraining. Without further experiments that cannot be determined. The network clearly has issues with animals, i.e. more organic shapes, but even a human may not be able to identify the butterfly at this resolution.

A collage of images of animals

Description automatically generated

### Discussion

ResNet, for the hyperparameters used, performed slightly better than VGG and would be expected in this case as ResNet employs skip connections to better deal with vanishing gradients, i.e. ResNet is a better head.

## Final Observations